**🡪How to create a deamon with shell scripting?**

**o how to run the above script?**  
**Ans :** Use **nohup** when running the script. For those people who don’t know nohup command here is the explnation.   
nohup is a command to run a program thought you logout from the machine. For example here is my script with nohup.

**nohup sh daemon.sh**

But some times this will not work. At that time run this script from **crontab** once and then remove the entry from crontab.   
**So how to make this permanent?**  
**Ans :**Its simple. keep your script in **/etc/init.d**with execute permissions

**cp /path/to/script/daemon.sh /etc/init.d/**  
**chmod +x /etc/init.d/daemon.sh**

Then create a link file to this script to the corresponding run-level. This is required at the time of booting.

**ln -s /etc/init.d/daemon.sh /etc/rc.d/rc3.d/S43daemon.sh**  
**ln -s /etc/init.d/daemon.sh /etc/rc.d/rc3.d/K43daemon.sh**

**What are those above two command will do?**  
**Ans :**The **S43** will tell the system to start the script as **43** script when it boots up.  
The **K43** will tell the system to **shutdown**cleanly when you do a shut down.

**How to Clear RAM Memory Cache, Buffer and Swap Space on Linux**

#### How to Clear Cache in Linux?

Every Linux System has three options to clear cache without interrupting any processes or services.

**1.** Clear PageCache only.

# sync; echo 1 > /proc/sys/vm/drop\_caches

**2.** Clear dentries and inodes.

# sync; echo 2 > /proc/sys/vm/drop\_caches

**3.** Clear PageCache, dentries and inodes.

# sync; echo 3 > /proc/sys/vm/drop\_caches

**sync** will flush the file system buffer.

#### How to Clear Swap Space in Linux?

If you want to clear Swap space, you may like to run the below command.

# swapoff -a && swapon -a

## Interpreting Free

To see how much memory you are currently using, run free -m.  It will provide output like:

:~$ free -m

total used free shared buffers cached

Mem: 2008 1951 57 0 142 575

-/+ buffers/cache: 1234 774

Swap: 3812 35 3777

The top row 'used' (1951) value will almost always nearly match the top row total value (2008). Since Linux likes to use any spare memory to cache disk blocks.

The most important 'used' figure to look at is the buffers/cache row used value (1234). This is how much space your applications are currently using. For best performance, this number should be less than your total (2008) memory. To prevent out of memory errors, it needs to be less than the total memory (2008) and swap space (3812).

If you wish to quickly see how much memory is free look at the buffers/cache row free value (774). This is the total memory (2008) - the actual used (1234). (2008 - 1234 = 774)

Note that the kernel does need some RAM for caching, to help improve system performance of some subsystems, for example disk writes. If you are seeing some slowness, and RAM looks ok but tight, adding more RAM (or reducing the amount used by applications) is a good idea.

# The GNU/Linux Kernel

Linux is the world-leading open-source kernel.

It is designed to peform well on a wide range of hardware.

## File Handle Limits

When you're serving a lot of traffic it is usually the case that the traffic you're serving is coming from a large number of local files.

The kernel has built-in limits on the number of files that a process can open, and raising these limits, at a cost of some system memory, is usually a sane thing to attempt.

You can view the current limit on the number of open-files by running:

$ cat /proc/sys/fs/file-max

The limit can be raised interactively by running, as root:

# sysctl -w fs.file-max=100000

If you wish that change to be made persistently you should append to the file /etc/sysctl.conf the line:

fs.file-max = 100000

Then run the following command to make your change take effect:

# sysctl -p

## Socket Tuning

For servers which are handling large numbers of concurent sessions, there are some TCP options that should probabaly be tweaked.

With a large number of clients comnunicating with your server it wouldn't be unusual to have a 20,000 open sockets or more. To increase that range you append the following to the bottom of /etc/sysctl.conf:

# Use the full range of ports.

net.ipv4.ip\_local\_port\_range = 1024 65535

You can also increase the recycling time of sockets, avoiding large numbers of them staying in the TIME\_WAIT status by adding these values to /etc/sysctl.conf:

# Enables fast recycling of TIME\_WAIT sockets.

# (Use with caution according to the kernel documentation!)

net.ipv4.tcp\_tw\_recycle = 1

# Allow reuse of sockets in TIME\_WAIT state for new connections

# only when it is safe from the network stack’s perspective.

net.ipv4.tcp\_tw\_reuse = 1

Finally one problem you'll find is that if a socket is listening and busy a connection-backlog will pile up. The kernel will keep pending connections in a buffer before failing. You can tweak several values to increase the size of the backlog:

#

# 16MB per socket - which sounds like a lot, but will virtually never

# consume that much.

#

net.core.rmem\_max = 16777216

net.core.wmem\_max = 16777216

# Increase the number of outstanding syn requests allowed.

# c.f. The use of syncookies.

net.ipv4.tcp\_max\_syn\_backlog = 4096

net.ipv4.tcp\_syncookies = 1

# The maximum number of "backlogged sockets". Default is 128.

net.core.somaxconn = 1024

The trade-off here is that a connecting client will see a slow connection, but this is almost certainly better than a Connection Refused error.

Once you've made those additions you can cause them to be loaded by running:

# sysctl -p

Finally if you've changed these limits you will need to restart the associated daemons. (For example "service nginx restart".)

## Process Scheduler

If you're running a recent ( newer than approx 2.6.32) you've got the 'Completely Fair Scheduler' (CFS) For modern systems serving lots of connections on lots of cores, you may hit issues with process migration.

There's a kernel parameter that determines how long a migrated process has to be running before the kernel will consider migrating it again to another core. The sysctl name is sched\_migration\_cost\_ns, default value 50000 (that's ns so 0.5 ms):

$ cat /proc/sys/kernel/sched\_migration\_cost\_ns

(It was renamed from sched\_migration\_cost at some point between 3.5 and 3.8)

Forking servers, like PostgreSQL or Apache, scale to much higher levels of concurrent connections if this is made larger, by at least an order of magnitude:

The limit can be raised interactively by running, as root:

# sysctl -w kernel.sched\_migration\_cost\_ns=5000000

If you wish that change to be made persistently you should append to the file /etc/sysctl.conf the line:

kernel.sched\_migration\_cost\_ns = 5000000

Another parameter that can dramatically impact forking servers is sched\_autogroup\_enabled. This setting groups tasks by TTY, to improve perceived responsiveness on an interactive system. On a server with a long running forking daemon, this will tend to keep child processes from migrating away as soon as they should. It can be disabled like so:

# sysctl -w kernel.sched\_autogroup\_enabled=0

Various PostgreSQL users have reported (on the postgresql performance mailing list) gains up to 30% on highly concurrent workloads on multi-core systems.

If you wish that change to be made persistently you should append to the file /etc/sysctl.conf the line:

kernel.sched\_autogroup\_enabled = 0

Then run the following command to make your change take effect:

# sysctl -p

## Filesystem Tuning

You almost certainly want to disable the "atime" option on your filesystems.

With this disabled that the last time a file was accessed won't be constantly updated every time you read a file, since this information isn't generally useful inand causes extra disk hits, its typically disabled.

To do this, just edit /etc/fstab and add "notime" as a mount option for the filesystem. For example:

/dev/rd/c0d0p3 /test ext3 noatime 1 2

## Swap Tuning

* **TODO**

## RAID Tuning

It seems to be the case that if you have the deadline scheduler this is best for RAID setups, however this is something that you'll want to test yourself.

Boot your kernel with elevator=deadline appended to the command-line and compare the result via your favourite [filesystem test](https://tweaked.io/benchmarking/#fs).

## Basic Questions

### ****1.**** ****What is initrd image and what is its function in the linux booting process ?****

The initial RAM disk (initrd) is an initial root file system that is mounted prior to when the real root file system is available.The initrd is bound to the kernel and loaded as part of the kernel boot procedure. The kernel then mounts this initrd as part of the two-stage boot process to load the modules to make the real file systems available and get at the real root file system. Thus initrd image plays a vital role in [linux booting process](http://linoxide.com/booting/boot-process-of-linux-in-detail/).

### ****2.**** ****Explain the terms suid, sgid and sticky bit ?****

In addition to the basic file permissions in Linux, there are few special permissions that are available for executable files and directories.

**SUID :** If setuid bit is set, when the file is executed by a user, the process will have the same rights as the owner of the file being executed.

**SGID :** Same as above, but inherits group previleges of the file on execution, not user previleges. Similar way when you create a file within directory,it will inherit the group ownership of the directories.

**Sticky bit** : Sticky bit was used on executables in linux so that they would remain in the memory more time after the initial execution, hoping they would be needed in the near future. But mainly it is on folders, to imply that a file or folder created inside a stickybit enabled folder could only be deleted by the owner. A very good implementation of sticky bit is /tmp ,where every user has write permission but only users who own a file can delete them.

### ****3.**** ****List out few of the differences between Softlink and Hardlink ?****

**a)** Hardlink cannot be created for directories. Hard link can only be created for a file.

**b)** Symbolic links or symlinks can link to a directory.

**c)** Removing the original file that your hard link points to does not remove the hardlink itself; the hardlink still provides the content of the underlying file.

**d)** If you remove the hard link or the symlink itself, the original file will stay intact.

**e)** Removing the original file does not remove the attached symbolic link or symlink, but without the original file, the symlink is useless

### ****4.**** ****How do you sent a mail attachment via bash console ?****

"mutt" is an opensource tool for sending emails with attachments from the linux bash command line. We can install "mutt" from the binary rpm or via packagemanager.

For Ubuntu / Debian based destros.

# apt-get install mutt

For Redhat / Fedor based destros,

# yum install mutt

Usage :

# mutt -s "Subject of Mail" -a "path of attachment file" "email address of recipient" < "message text containing body of the message"

Eg : mutt -s "Backup Data" -a /home/backup.tar.gz admin@mywebsite.com < /tmp/message.txt

### ****5.**** ****What is the difference between umask and ulimit ?****

umask stands for ‘User file creation mask’, which determines the settings of a mask that controls which file permissions are set for files and directories when they are created. While ulimit is a linux built in command which provides control over the resources available to the shell and/or to processes started by it.

You can limit user to specific range by editing /etc/security/limits.conf at the same time system wide settings can be updated in /etc/sysctl.conf

### ****6.**** ****What are the run levels in linux and how to change them ?****

A run level is a state of init and the whole system that defines what system services are operating and they are identified by numbers.There are 7 different run levels present (run level 0-6) in Linux system for different purpose. The descriptions are given below.

0: Halt System (To shutdown the system)  
1: Single user mode  
2: Basic multi user mode without NFS  
3: Full multi user mode (text based)  
4: unused  
5: Multi user mode with Graphical User Interface  
6: Reboot System

To change the run level, edit the file “/etc/inittab” and change initdefault entry ( id:5:initdefault:). If we want to change the run level on the fly, it can be done using ‘init’ command.

For example, when we type ‘init 3' in the commandline , this will move the system from current runlevel to runlevl 3. Current level can be listed by typing the command 'who -r'

### ****7.**** ****What is the functionality of a Puppet Server ?****

Puppet is an open-source and enterprise application for configuration management toll in UNIX like operating system. Puppet is an IT automation software used to push configuration to its clients (puppet agents) using code. Puppet code can do a variety of tasks from installing new software, to check file permissions, or updating user accounts and lots of other tasks.

### ****8.**** ****What is SeLinux?****

SELinux is an acronym for Security-enhanced Linux. It is an access control implementation and security feature for the Linux kernel. It is designed to protect the server against misconfigurations and/or compromised daemons. It put limits and instructs server daemons or programs what files they can access and what actions they can take by defining a security policy.

### ****9.**** ****What is crontab and explain the fields in a crontab ?****

The cron is a deamon that executes commands at specific dates and times in linux. You can use this to schedule activities, either as one-time events or as recurring tasks. Crontab is the program used to install, deinstall or list the tables used to drive the cron daemon in a server. Each user can have their own crontab, and though these are files in /var/spool/cron/crontabs, they are not intended to be edited directly. Here are few of the command line options for crontab.

crontab -e Edit your crontab file.  
crontab -l Show your crontab file.  
crontab -r Remove your crontab file.

Traditional cron format consists of six fields separated by white spaces:

<Minute> <Hour> <Day\_of\_the\_Month> <Month\_of\_the\_Year> <Day\_of\_the\_Week> <command/program to execute>

The format is explained in detail below.

\* \* \* \* \* \*  
| | | | | |  
| | | | | +-- Year (range: 1900-3000)  
| | | | +---- Day of the Week (range: 1-7, 1 standing for Monday)  
| | | +------ Month of the Year (range: 1-12)  
| | +-------- Day of the Month (range: 1-31)  
| +---------- Hour (range: 0-23)  
+------------ Minute (range: 0-59)

### ****10.**** ****What are inodes in Linux ? How to find the inode associated with a file ?****

An inode is a data structure on a filesystem on Linux and other Unix-like operating systems that stores all the information about a file except its name and its actual data. When a file is created, it is assigned both a name and an inode number, which is an integer that is unique within the filesystem. Both the file names and their corresponding inode numbers are stored as entries in the directory that appears to the user to contain the files. The concept of inodes is particularly important to the recovery of damaged filesystems. When parts of the inode are lost, they appear in the lost+found directory within the partition in which they once existed.

The following will show the name of each object in the current directory together with its inode number:

# ls -i

The avialble number inodes in a filesystem can be found using the below command :

# df -i

The other way we can get the inode details of a file by using the stat commmand.

Usage : # stat <file name>

Example :

-sh-4.1$ stat note.txt  
File: `note.txt'  
Size: 4 Blocks: 8 IO Block: 4096 regular file  
Device: fd05h/64773d Inode: 8655235 Links: 1  
Access: (0644/-rw-r--r--) Uid: (69548/nixuser) Gid: (25000/ UNKNOWN)  
Access: 2014-06-29 15:27:56.299214865 +0000  
Modify: 2014-06-29 15:28:28.027093254 +0000  
Change: 2014-06-29 15:28:28.027093254 +0000

Apart from the above basic questions, be prepared for answers for the below questions

**1.** How to set linux file/directory permissions ?

**2.** How to set ownership for files/directories ?

**3.** How to create user/group and how to modify it ?

**4.** How to find kernel / OS version and its supported bit (32/64) version ?

**5.** How to set / find interface ip address ?

**6.** How to find linux mount points and disk usage ?

**7.** What command to find memory and swap usage ?

**8.** Have a look on ps, top, grep, find, awk and dmesg commands ?

## Linux Scenario Questions

### ****11.**** ****What is the difference between name based virtual hosting and IP based virtual hosting ? Explain the scenario where name based virtual hosting seems useful ?****

Virtual hosts are used to host multiple domains on a single apache instance. You can have one virtual host for each IP your server has, or the same IP but different ports, or the same IP, the same port but different host names. The latter are called "name based vhosts".

n IP-based virtual hosting, we can run more than one web site on the same server machine, but each web site has its own IP address while In Name-based virtual hosting, we host multiple websites on the same IP address. But for this to succeed, you have to put more than one DNS record for your IP address in the DNS database.

In the production shared webhosting environment, getting a dedicated IP address for every domains hosted in the server is not feasible in terms of cost. Most of the customers wont be able to afford the cost of having a dedicated IP address. Here is the place where the concepts of Name based virtual hosting find its place.

### ****12.**** ****What is network bonding in Linux and where the important configuration files involved? What is the advantage of Network Bonding ?****

Network Bonding is a Linux kernel feature that allows to aggregate multiple network interfaces into a single virtual link. This is a great way to achieve redundant links, fault tolerance or load balancing networks in production system. If one of the physical NIC is down or unplugged, it will automatically move traffic to the other NIC card. Similar way the bonding will increase the interface throughput to handle the traffic it it is configured in active-active mode.

There are 7 modes starting from 0 to 6 which decides how the bonding configuration behaves.

**mode=0** (balance-rr) - Round-robin policy

It the default mode. It transmits packets in sequential order from the first available slave through the last.  
This mode provides load balancing and fault tolerance.

**mode=1** (active-backup)

Active-backup policy: In this mode, only one slave in the bond is active. The other one will become active, only when the active slave fails. The bond’s MAC address is externally visible on only one port (network adapter) to avoid confusing the switch. This mode provides fault tolerance.

**mode=2** (balance-xor)

Transmit the traffic based on [(source MAC address XOR'd with destination MAC address) modulo slave count]. This selects the same slave for each destination MAC address. This mode provides load balancing and fault tolerance.

**mode=3** (broadcast)

Broadcast policy: transmits everything on all slave interfaces. This mode provides fault tolerance.

**mode=4** (802.3ad)

Creates aggregation groups that share the same speed and duplex settings. Utilizes all slaves in the active aggregator according to the 802.3ad specification.

**mode=5** (balance-tlb) - Adaptive transmit load balancing

channel bonding that does not require any special switch support. The outgoing traffic is distributed according to the current load (computed relative to the speed) on each slave. Incoming traffic is received by the current slave. If the receiving slave fails, another slave takes over the MAC address of the failed receiving slave.

**mode=6** (balance-alb) - Adaptive load balancing

It includes balance-tlb plus receive load balancing (rlb) for IPV4 traffic, and does not require any special switch support. The receive load balancing is achieved by ARP negotiation.

Important Configuration Files involved :

/etc/sysconfig/network-scripts/ifcfg-bond0  
/etc/modprobe.d/bonding.conf  
/etc/sysconfig/network-scripts/ifcfg-eth[0-4]  
/proc/net/bonding/bond0

### ****13.**** ****Explain briefly the procedure for re-installing Grub in Linux ?****

**1)**Download Ubuntu Installation / Live cd

**2)**Boot from Ubuntu Installation / Live cd - usb, burned cd etc.  
**3)**During boot select "Try Ubuntu" , Don't select install !  
**4)** Mount your Linux root partition  
sudo mount /dev/sda6 /mnt ( Assuming /dev/sda6 is the Linux root partition)  
**5)** Install / reinstall grub  
$ sudo grub-install --root-directory=/mnt/ /dev/sda ( where /dev/sda is your primary disk)  
Installation finished. No error reported.

**6)** Reboot your system, remove bootable CD and we should have the boot menu ready when the system starts.

Note : There would be slight difference when using with other distros.

### ****14.**** ****Explain the fields in /etc/passwd and /etc/shadow ?****

The /etc/shadow file stores actual password in encrypted format with some additional properties related to user password.It mainly holds athe account aging parameters. All fields are separated by a colon (:) symbol. It contains one entry per line for each user listed in /etc/passwd file Generally, shadow file entry looks as below.

steve:$1$XOdE07rn$WA6qFm4W5UIqNfaqE5Uub.:13775:0:99999:7:::

Here is the explanation of each field.

**User name** : Your login name

**Password**: Your encrypted password.

**Last password change** : Days since Jan 1, 1970 that password was last changed

**Minimum**: The minimum number of days required between password changes.

**Maximum**: The maximum number of days the password is valid.

**Warn** : The number of days before password is to expire that user is warned that his/her password must be changed

**Inactive** : The number of days after password expires that account is disabled

**Expire** : days since Jan 1, 1970 that account is disabled. It indicates an absolute date specifying when the login may no longer be used

The /etc/passwd file stores essential information, which is required during login /etc/passwd is a text file, that contains a list of user account related parameters like user ID, group ID, home directory, shell, etc.

Here is the sample entry from /etc/passwd file

steve:x:6902:6902::/home/steve:/bin/bash

**Username**: User's login name.

**Password**: An x character indicates that encrypted password is stored in /etc/shadow file.

**User ID (UID)**: Each user must be assigned a user ID (UID). UID 0 (zero) is reserved for root.

**Group ID (GID)**: The primary group ID

**User Info**: The comment field. It allow you to add extra information about the user.

**Home directory**: The absolute path to the directory the user will be in when they log in.

**Command/shell**: The absolute path of a command or shell (/bin/bash).

### ****15.**** ****How do you boot your system into the following modes, when you are in some trouble ?****

**a)** Rescue mode  
**b)** Single user mode  
**c)** Emergency mode

Rescue mode provides the ability to boot a small Linux environment from an external bootable device like a CD-ROM, or USB drive instead of the system's hard drive.Rescue mode is provided to help you with your system from repairing the file system or fixing certain issues which prevent your normal operations.

In order to get into the rescue mode, change the BIOS settings of the machine to boot from the external media. Once the system started booting using bootable disk, add the keyword rescue as a kernel parameter or else you can give the parameter "linux rescue" in the graphical boot interface.

In single-user mode, the system boots to runlevel 1, but it will have many more additional functionalities compared to switching to runlevel 1 from other levels.

The local file systems can be mounted in this mode, but the network is not activated.

Use the following steps to boot into single-user mode:

**1)**At the GRUB splash screen during the booting process, press any key to enter the GRUB interactive menu.  
**2)**Select the proper version of kernel that you wish to boot and type "a" to append the line.  
**3)**Go to the end of the line and type "single" as a separate word.  
**4)**Press Enter to exit edit mode and type "b" to boot into single usermode now.

In emergency mode, you are booting into the most minimal environment possible. The root file system is mounted read-only and almost nothing is set up. The main advantage of emergency mode over single-user mode is that the init files are not loaded. If the init is corrupted , you can still mount file systems to recover data that could be lost during a re-installation. To boot into emergency mode, use the same method as described for single-user mode, with one exception, replace the keyword single with the keyword "emergency".

### ****16.**** ****In the ps results few of the processes are having process state as "D" . What does it mean ? Briefly explain different process states ?****

To have a dynamic view of a process in Linux, always use the top command. This command provides a real-time view of the Linux system in terms of processes. The eighth column in the output of this command represents the current state of processes. A process state gives a broader indication of whether the process is currently running, stopped, sleeping etc.

A process in Linux can have any of the following four states…

**Running** – A process is said to be in a running state when either it is actually running/ executing or waiting in the scheduler’s queue to get executed (which means that it is ready to run). That is the reason that this state is sometimes also known as ‘runnable’ and represented by (R).

**Waiting or Sleeping** – A process is said to be in this state if it is waiting for an event to occur or waiting for some resource-specific operation to complete. So, depending upon these scenarios, a waiting state can be subcategorised into an interruptible (S) or uninterruptible (D) state respectively.

**Stopped** – A process is said to be in the stopped state when it receives a signal to stop. This usually happens when the process is being debugged. This state is represented by (T).

**Zombie** – A process is said to be in the zombie state when it has finished execution but is waiting for its parent to retrieve its exit status. This state is represented by (Z).

Apart from these four states, the process is said to be dead after it crosses over the zombie state; ie when the parent retrieves its exit status. ‘Dead’ is not exactly a state, since a dead process ceases to exist.

### ****17.**** ****What is drop cache in Linux and how do you clear it ?****

Cache in Linux memory is where the Kernel stores the information it may need later, as memory is incredible faster than disk.

It is great that the Linux Kernel takes care about that.Linux Operating system is very efficient in managing your computer memory, and will automatically free the RAM and drop the cache if some application needs memory.

Kernels 2.6.16 and newer provide a mechanism to have the kernel drop the page cache and/or inode and dentry caches on command, which can help free up a lot of memory. Now we can throw away that script that allocated a ton of memory just to get rid of the cache.

To free pagecache:

# echo 1 > /proc/sys/vm/drop\_caches

To free dentries and inodes:

# echo 2 > /proc/sys/vm/drop\_caches

To free pagecache, dentries and inodes:

echo 3 > /proc/sys/vm/drop\_caches

This is a non-destructive operation in normal scenarios and will only free things that are completely unused. Dirty objects will continue to be in use until written out to disk and are not freeable. However it is always preferred to run "sync" first to flush useful things out to disk.

### ****18.**** ****Password based authentication is disabled in your infrastructure. So how do you login to the servers ?****

To improve the system security even further, most of the organizations turned to use keybased authentications instead of Password based authentication. We can enforce the key-based authentication by disabling the standard password authentication, which involves a public key private key pair. The public key is added in the server configuration file while private key is kept kept confidential on the client side.

Below listed is the procedure, to set up keybased authentication.

**1)** Generating Key Pairs

a) Generate an RSA key pair by typing the following at a shell prompt:

$ ssh-keygen -t rsa  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/steve/.ssh/id\_rsa):

b) Press Enter to confirm the default location (that is, ~/.ssh/id\_rsa) for the newly created key.

c) Enter a passphrase, and confirm it by entering it again when prompted to do so.

d) Copy the content of ~/.ssh/id\_rsa.pub into the ~/.ssh/authorized\_keys on the machine to which you want to connect,  
appending it to its end if the file already exists.

e) Change the permissions of the ~/.ssh/authorized\_keys file using the following command:

$ chmod 600 ~/.ssh/authorized\_keys

**2)** Now on your client side, open the remote connection agent like putty and browse your public key and try SSH to the server, you should be able to login without a password now.

# ssh server1.myserver.com  
The authenticity of host 'server1.myserver.com (192.168.44.2)' can't be established.  
RSA key fingerprint is e3:c3:89:37:4b:94:37:d7:0c:d5:6f:9a:38:62:ce:1b.  
Are you sure you want to continue connecting (yes/no)? yes  
Warning: Permanently added 'server1.myserver.com' (RSA) to the list of known hosts.  
Last login: Tue July 13 12:40:34 2014 from server2.myserver.com

**3)** Public key authentication can prevent brute force SSH attacks, but only if all password-based authentication methods are disabled. Once public key authentication has been confirmed to be working, disable regular password authentication by editing /etc/ssh/sshd\_config and set the following option to "no".

PasswordAuthentication no

### ****19.**** ****Explain the different Scenarios involved in TCP 3 way handshake ?****

The TCP three way handshake is the process for establishing a TCP connection.We can explain 3 way handshake with a simple scenario where we assume a client computer is contacting a server to send it some information.

**a)** The client sends a packet with the SYN bit set and a sequence number of N.  
**b)** The server sends a packet with an ACK number of N+1, the SYN bit set and a sequence number of X.  
**c)** The client sends a packet with an ACK number of X+1 and the connection is established.  
**d)** The client sends the data.

The first three steps in the above process is called the three way handshake.

### ****20.**** ****As the disk space utilization was so high in the server, the Administrator has removed few files from the server but still the disk utilization is showing as high. What would be the reason ?****

In Linux even if we remove a file from the mounted file system, that will still be in use by some application and for this application it remains available. Its because file descriptor in /proc/ filesystem is held open..So if there are such open descriptors to files already removed, space occupied by them considered as used. You find this difference by checking them using the "df" and "du" commands. While df is to show the file system usage, du is to report the file space usage. du works from files while df works at filesystem level, reporting what the kernel says it has available.

You can find all unlinked but held open files with:

# lsof | grep '(deleted)'

This will list the filename which is open with the pid in which it is running. We can kill those Pids and which will stop these process and will recover the disk space responsible for this file.

### ****21.**** ****What is rDNS and explain its benefits in the Linux Domain Name Systems ?****

A typical DNS lookup is used to determine which IP address is associated with a hostname, and this is called Forward DNS lookup. A reverse DNS lookup is used for the opposite, to determine which hostname is associated with an IP address. Sometimes reverse DNS lookups are required for diagnostic purposes. Today, reverse DNS lookups are used mainly for security purposes to trace a hacker or spammer. Many modern mailing systems use reverse mapping to provide simple authentication using dual lookup: hostname-to-address and address-to-hostname. The rDNS ( reverse DNS ) is implemented using a specialized zone record for reverse lookups called PTR record. PTR records always resolve to names, never IP addresses.

### ****22.**** ****What is sosreport, how do you generate it while working with your Redhat Support Team in production ?****

Sosreport is a command-line utility in Redhat based linux destros (RHEL / CentOS) which collects system configuration and diagnostic information of your linux box like running kernel version, loaded modules, and system and service configuration files. This command also runs external programs to collect further information, and stores this output in the resulting archive. Sosreport is required when you have open a case with redhat for technical support. Redhat support Engineers will require sosreport of your server for troubleshooting purpose. To run sosreport, sos package should be installed. Sos package is part of default installation in most of linux. If for any reason this package is no installed , then use below yum command to install it manually :

# yum install sos

Generate the report

Open the terminal type sosreport command :

# sosreport

This command will normally complete within a few minutes. Depending on local configuration and the options specified in some cases the command may take longer to finish. Once completed, sosreport will generate a compressed a file under /tmp folder. The file should be provided to Redhat support representative as an attachment to open a support case.

### ****23.**** ****What is swappiness in Linux Memory Management and how do we configure that ?****

The swappiness parameter controls the tendency of the kernel to move processes out of physical memory and onto the swap disk. Because disks are much slower than RAM, this can lead to slower response times for system and applications if processes are too aggressively moved out of memory.

swappiness can have a value of between 0 and 100

swappiness=0 tells the kernel to avoid swapping processes out of physical memory for as long as possible

swappiness=100 tells the kernel to aggressively swap processes out of physical memory and move them to swap cache

The default setting in Redhat/Ubuntu based Linux distros is swappiness=60. Reducing the default value of swappiness will probably improve overall performance for a typical Ubuntu desktop installation.

~$ cat /proc/sys/vm/swappiness  
60

If we have enough RAM, we can turn that down to 10 or 15. The swap file will then only be used when the RAM usage is around 80 or 90 percent.

To change the system swappiness value, open /etc/sysctl.conf as root. Then, change or add this line to the file:

vm.swappiness = 10

Reboot for the change to take effect

You can also change the value while your system is still running

sysctl vm.swappiness=10

We can also clear swap by running swapoff -a and then swapon -a as root instead of rebooting to achieve the same effect.

### ****24.**** ****What is git ?****

Git is a very popular and efficient open source Version Control System. It tracks content such as files and directories. It stores the file content in BLOBs - binary large objects. The folders are represented as trees. Each tree contains other trees (subfolders) and BLOBs along with a simple text file which consists of the mode, type, name and Secure Hash Algorithm of each blob and subtree entry. During repository transfers, even if there are several files with the same content and different names, the GIT software will transfer the BLOB once and then expand it to the different files.

### ****25.**** ****What is inode ? Briefly explain the structure ?****

Inode is a data structure that keeps track of all the information about a file. When we keep our information in a file and the OS stores the information about a file in an inode. Information about files is sometimes called metadata. We can say that an inode is metadata of the data. In a file system, inodes consist roughly of 1% of the total disk space, whether it is a whole storage unit or a partition on a storage unit. The inode space is used to ?track? the files stored on the hard disk. The inode entries store metadata about each file, directory or object, but only points to these structures rather than storing the data. Each entry is 128 bytes in size. The metadata contained about each structure can include the following:

Inode number  
Access Control List (ACL)  
Extended attribute  
Direct/indirect disk blocks  
Number of blocks  
File access, change and modification time  
File deletion time  
File generation number  
File size  
File type  
Group  
Number of links  
Owner  
Permissions  
Status flags

Inode structure of a directory consists of a name to inode mapping of files and directories in that directory.In a directory, You can find the inode number corresponding to the files using the command "ls -i"

786727 -rw------- 1 root root 4226530 May 29 13:17 sudo.log  
786437 -rw-------. 1 root root 32640 Jun 23 20:11 tallylog  
786440 -rw-rw-r--. 1 root utmp 276096 Jul 20 06:45 wtmp  
786741 -rw------- 1 root root 9653 Jul 17 09:38 yum.log

Similar way, the number of inodes allocated, used and free in a Filesystem can be listed using "df -i" command

# df -i /root  
Filesystem Inodes IUsed IFree IUse% Mounted on  
/dev/mapper/RootVol-lvmroot  
524288 80200 444088 16% /

### What is monolithic kernel?

A monolithic kernel is [set up](javascript:void(0);) so that all the drivers are compiled in the kernel. There's no need for the user to load any modules. However when you install new drivers, you need to recompile the kernel. As a result kernel becomes larger and slower ans requires more memory.

### What is modular kernel?

A modular kernel is set up so that most hardware drivers are compiled as modules. As a result you don't have to recompile the kernel when installing new hardware devices.

**Some bullet points**

The Linux kernel acts as a [mediator](javascript:void(0);) between system's hardware and software.

It is also responsible for memory management and multiplexing

It divides a [CPUs](javascript:void(0);) processing capacity between application so that they can run simultaneously

# [What is Initrd Image ? How To Create Initrd Image In Linux](http://linoxide.com/linux-how-to/initrd-image-linux/)

The boot process of a Linux system involves a number of stages. These include the BIOS initialization, reading the MBR, the bootloader, kernel initialization and the init process etc. The initrd, initial ramdisk plays a very significant role in booting up the system. This article discusses this initrd ramdisk and how we can create it if it needs to be recreated.

### GRUB bootloader and initrd

A bootloader is the first software program that runs during the startup process. It is responsible for booting the operating system. Bootloader loads the kernel into the memory and kernel mounts the root partition so that it can start the first process, i.e. init process. Let us have a look at the commands that the bootloader executes for these tasks:

root (hd0,0)  
kernel /vmlinuz-2.6.18-238.el5 ro root=/dev/VolGroup00/LogVol00  
initrd /initrd-2.6.18-238.el5.img

The first command "root (hd0,0)" tells the bootloader that kernel image is present on first partition of the hard disk hd0 (i.e. hd0,0). This is the location of the kernel image.

The second command is the kernel image itself. In this case, it is "/vmlinuz-2.6.18-238.el5". The arguments passed to this command are ro and root. ro specifies the read only mode and root tells the partition on which root filesystem resides. The kernel mounts this root partition in read-only mode.  
The third command is the location of initrd. Now to understand what initrd is, let’s look at an issue that the kernel faces while mounting the root partition.

### Explain About Grub.conf file in Redhat Linux Server

Grub.conf file it has 7 lines. there default file format.

timeout=5

splashimage=(hd0,0)/grub/splash.xpm.

title Red Hat Enterprise Linux Server (2.6.18-53.el5)

root (hd0,0)

kernel /vmlinuz-2.6.18-53.el5 **roroot=/dev/VolGroup00/LogVol00**

initrd /initrd-2.6.18-53.el5.img

**FIRST LINE:** timeout=5  
while the system boot it will wait up to 5 second user to select the kernel.  
  
**SECOND LINE:** splashimage=(hd0,0)/grub/splash.xpm.  
it will load the (hd0,0)/grub/splash.xpm.  image as the background when we were booting the OS  
**How to create teh splash.xpm.gz Image**  
#convert imagelocation.png -resize 640x480 -colors 14 splash.xpm  
#gzip splash.xpm  
  
**THIRD LINE**:title Red Hat Enterprise Linux Server (2.6.18-53.el5)  
It will give the kernel label as the “title Red Hat Enterprise Linux Server (2.6.18-53.el5)”  
  
**FOURTH LINE**:root (hd0,0)  
Here (hdx,y) x -> Hard Drive y-> Partition  
we need to give the Correct partition  
otherwise it will through following error message  
Error 22: No such partition  
  
**FIFTH LINE**:kernel /vmlinuz-2.6.18-53.el5 ro root=**/dev/VolGroup00/LogVol00** rhgb quiet  
it will load the kernel image and mount root file system as mention in**root=**  
/dev/VolGroup00/LogVol00 refer to the logical volume root file system located in these logical volume.  
And kernal execute /sbin/init process it is parent process PID is 1.  
  
**SIX LINE:**initrd /initrd-2.6.18-53.el5.img  
initrd it has executable driver module.  
initrd Intilize the Ram Disk  
if we face the any problem to load initrd we need to reinstall by using mkinitrd.

### The Chicken Egg Module Problem

We just saw that the kernel has to mount the root partition in read-only mode. But the root filesystem can be on a partition with one or more of following capabilities:

• Logical Volume Management (as in our case)  
• Software RAID  
• NFS  
• Encrypted partition  
• SCSI Controller Support

The kernel needs modules for these devices, as they are not compiled into kernel itself. So kernel will load modules for these devices at runtime. These modules are present in “/lib/modules/” directory. This directory is present on the root filesystem itself. And the root partition is not mounted yet (that is what this all is about). So how can kernel access the modules for mounting the root partition residing on the root partition itself (without mounting it)?

Here initrd comes to rescue. The initrd ramdisk contains the modules required for mounting the root partition. This initrd resides on the same partition on which kernel image is present. So the kernel loads the initrd in memory, accesses the modules and mounts the root partition in read-only mode.

### Creating initrd

When the filesystem hardware or software changes, then we need to recreate the initrd. initrd can be created with “mkinitrd” command. The location of initrd is /boot directory. The kernel version for which the initrd image is being created needs to be passed as an argument to the mkinitrd command. The current kernel version can be checked with "uname -r" command:

[root@redhat-server ~]# uname -r  
2.6.18-238.el5

We can use this command to pass the required argument with command substitution:

[root@redhat-server ~]# mkinitrd /boot/initrd-latest.img $(uname -r)

The $(uname -r) will substitute the output of the command "uname -r" in its place.

The initrd image is a compressed image. You can check this with file command:

[root@redhat-server ~]# file /boot/initrd-latest.img  
/boot/initrd-latest.img: gzip compressed data, from Unix, last modified: Fri Aug 3 10:47:47 2012, max compression

### [What is GRUB boot Loader ?](http://www.golinuxhub.com/2014/03/what-is-grub-boot-loader.html)

POSTED BY DEEPAK PRASAD SUNDAY, MARCH 16, 2014 [NO COMMENTS](http://www.golinuxhub.com/2014/03/what-is-grub-boot-loader.html#comment-form)

On a Red Hat Linux system, the boot loader's function is to locate the Linux Kernel, and any other necessary files, and load them into [memory](javascript:void(0);). It then starts the kernel so that the kernel can run processes.  
  
The boot loader also lets you to control how a system is booted. If you dual boot, a boot loader enables you to choose between [operating systems](javascript:void(0);) at startup.  
  
When a computer boots, the basic input/output system (BIOS) in read only memory usually loads the boot loader from the Master Boot Record (MBR) on the system's primary hard drive.  
  
However some versions of Linux run directly from the flash memory, by passing the BIOS and the boot loader. Linux was designed to run without an interactive boot loader with the kernel located at particular sectors on the disk.  
  
**Using a boot loader offers several benefits including**

* the ability to boot multiple operating systems
* the ability to pass parameters interactively to the kernel, which is useful for disabling certain features in order to solve hardware problems
* the ability to load different kernels interactively, which is useful when you deploy a new kernel because it is easy to revert to the working kernel.

### What are the stages involved in loading GRUB?

Most boot loaders execute in two or more stages. GRUB loads itself into memory by  
  
**1. loading the stage 1 boot loader**

* First the BIOS reads the stage 1 or primary boot loader into memory from the MBR.
* The primary boot loader takes up less than 512 bytes of [disk space](javascript:void(0);) in the MBR - too small a space to contain the instructions necessary to load a complex [operating system](javascript:void(0);).
* Instead the primary boot loader performs the function of loading either the stage 1.5 or stage 2 boot loader.

**2. loading the stage 1.5 boot loader**

* Some hardware requires an intermediate step between the stage 1 and stage 2 boot loaders.
* This can happen when the /boot [partition](javascript:void(0);) is situated beyond the 1024 cylinder head of the hard drive, or when you are using LBA mode.
* GRUB Stage 1.5 is located in the first 30 KB of Hard disk immediately after MBR and before the first partition.
* This space is utilised to store file system drivers and modules.
* This enabled stage 1.5 to load stage 2 to load from any known loaction on the file system i.e. /boot/grub

**3. loading the stage 2 boot loader**

* The secondary boot loader is located somewhere on the disk - on the boot sector of the first partition.
* For Example it displays the GRUB menu and command environment, which enables you to select which operating system or Linux kernel to boot.
* You can also use it to pass arguments to the kernel.

**4. loading the operating system or kernel**

* GRUB determines which operating system to start and loads the operating system or kernel and initrd into memory.
* It then transfers control of [the computer](javascript:void(0);) to the OS.

### GRUB vs LILO

The two boot loaders that Red Hat Linux provides for x86 architecture are the Grand Unified Boot Loader (**GRUB**) and the Linux Loader (**LILO**). In the past LILO has been the most popular for Linux and the default loader on most distributions.  
  
However, the GNU project's GRUB is new more commonly used.  
  
GRUB is currently the default Red Hat Linux boot Loader. It is an improvement over LILO in that it enables you to

* use [password protection](javascript:void(0);)
* try different boot parameters during the boot process without permanently hanging the GRUB configuration file.
* boot Windows NT operating system from the MBR area of the hard drive

GRUB also supports Logical Block Addressing (**LBA**) mode which enables the computer to find the /boot files more easily, particularly if they are located beyond the 1024th cylinder of the hard drive.

### Direct vs Chain-loading Method

The boot method for Red Hat Linux is called the **direct loading method**, because the boot loader loads the operating system directly without using an intermediary stage between the boot loader and the kernel.  
  
In the **chain loading** boot method, the MBR points to the first sector of the partition that contains the operating system. The MBR uses the files in this location to boot the operating system.  
  
GRUB supports both direct and chain loading boot methods, so it can boot almost any operating system.

 dd Command

### Syntax of ‘dd’ command

dd  if=<source file name> of=<target file name> [Options]

Copy a file, converting and formatting according to the operands:

**bs=BYTES read and write up to BYTES bytes at a time**

**cbs=BYTES convert BYTES bytes at a time**

**conv=CONVS convert the file as per the comma separated symbol list**

**count=N copy only N input blocks**

**ibs=BYTES read up to BYTES bytes at a time (default: 512)**

**if=FILE read from FILE instead of stdin**

**iflag=FLAGS read as per the comma separated symbol list**

**obs=BYTES write BYTES bytes at a time (default: 512)**

**of=FILE write to FILE instead of stdout**

**oflag=FLAGS write as per the comma separated symbol list**

**seek=N skip N obs-sized blocks at start of output**

**skip=N skip N ibs-sized blocks at start of input**

**status=WHICH WHICH info to suppress outputting to stderr;**

**'noxfer' suppresses transfer stats, 'none' suppresses all**

### Numerical Suffixes

**N** and **BYTES** may be followed by the following multiplicative suffixes:  
  
**c**=1  
**w**=2  
**b**=512  
**kB**=1000  
**K**=1024  
**MB**=1000\*1000  
**M**=1024\*1024  
**xM**=M  
**GB**=1000\*1000\*1000  
**G**=1024\*1024\*1024  
  
and so on for **T** (Terabytes), **P** (petabytes), **E** (exabytes), **Z** (zettabytes), and **Y** (yottabytes).  
  
  
Each **CONV** symbol may be:

**ascii from EBCDIC to ASCII**

**ebcdic from ASCII to EBCDIC**

**ibm from ASCII to alternate EBCDIC**

**block pad newline-terminated records with spaces to cbs-size**

**unblock replace trailing spaces in cbs-size records with newline**

**lcase change upper case to lower case**

**ucase change lower case to upper case**

**sparse try to seek rather than write the output for NUL input blocks**

**swab swap every pair of input bytes**

**sync pad every input block with NULs to ibs-size; when used**

**with block or unblock, pad with spaces rather than NULs**

**excl fail if the output file already exists**

**nocreat do not create the output file**

**notrunc do not truncate the output file**

**noerror continue after read errors**

**fdatasync physically write output file data before finishing**

**fsync likewise, but also write metadata**

Each **FLAG** symbol may be:

**append append mode (makes sense only for output; conv=notrunc suggested)**

**direct use direct I/O for data**

**directory fail unless a directory**

**dsync use synchronised I/O for data**

**sync likewise, but also for metadata**

**fullblock accumulate full blocks of input (iflag only)**

**nonblock use non-blocking I/O**

**noatime do not update access time**

**nocache discard cached data**

**noctty do not assign controlling terminal from file**

**nofollow do not follow symlinks**

**count\_bytes treat 'count=N' as a byte count (iflag only)**

**skip\_bytes treat 'skip=N' as a byte count (iflag only)**

**seek\_bytes treat 'seek=N' as a byte count (oflag only)**

**Other Options** are:

**--help display this help and exit**

**--version output version information and exit**

##### **rsync command**

# rsync options source destination

1. -v : verbose
2. -r : copies data recursively (but don’t preserve timestamps and permission while transferring data
3. -a : archive mode, archive mode allows copying files recursively and it also preserves symbolic links, file permissions, user & group ownerships and timestamps
4. -z : compress file data
5. -h : human-readable, output numbers in a human-readable format

##### **Copy a File from a Remote Server to a Local Server with SSH**

To specify a protocol with **rsync** you need to give “**-e**” option with protocol name you want to use. Here in this example, We will be using “**ssh**” with “**-e**” option and perform data transfer.

[root@tecmint]# rsync -avzhe ssh root@192.168.0.100:/root/install.log /tmp/

Command Syntax Checking

|  |  |
| --- | --- |
| **Daemon** | **Command** |
| OpenSSH | /usr/sbin/sshd -t && echo $? /usr/sbin/sshd -T |
| Apache | /usr/sbin/apache2 -t apachectl configtest |
| nginx | /usr/local/nginx/sbin/nginx -t /usr/local/nginx/sbin/nginx -t -c /usr/local/nginx/conf/nginx.conf |
| lighttpd | /usr/local/sbin/lighttpd -t -f /usr/local/etc/lighttpd/cyberciti.biz/lighttpd.conf |
| Bind (named server config) | named-checkconf /etc/named.conf |
| Bind (zone syntx) | named-checkzone cyberciti.biz /var/named/zone.cyberciti.biz |
| Squid proxy | /usr/sbin/squid -k check /usr/sbin/squid -k parse |
| MySQL server | mysqld --verbose --help /usr/libexec/mysqld --verbose --help 1>/dev/null |
| Postfix MTA | postfix check postfix -vvv |
| Samba SMB/CIFS | testparm -v |
| tcpd | tcpdchk tcpdchk -v |
| dhcpd (DHCP / BOOTP) server | dhcpd -t -cf /path/to/dhcpd.testing.conf |
| vsftpd server | vsftpd -olisten=NO /path/to/vsftpd.testing.conf |
| nagios | nagios -v /path/to/testing/nagios.cfg |
| Openntpd NTPD server | ntpd -d -f /usr/local/etc/ntpd.conf -n |
| Xorg (X11 Server) | Xorg -config /path/to/xorg.conf.new -retro |
| syslogd / rsyslogd | rsyslogd -c4 -f /etc/rsyslog.testing.conf -N 1 |
| CUPS Printing System | cupsd -f -c /path/to/cupsd.testing.conf -t |
| slapd (OpenLDAP) | slapd -Tt |
| varnishd | varnishd -C -f /path/to/wordpress.vlc |
| exim MTA | exim -bV |
| Bash/Ksh scripts | bash -n ./myscript ksh -n /path/to/script.ksh |
| BSD pf firewall | pfctl -nf /etc/pf.conf |
| proftpd | proftpd -t -c /path/to/proftpd.testing.conf |
| Perl scripts | perl -c /path/to/script.pl perl -wc /path/to/script.pl |

CentOS Find Out Latest Available Updated Package or Kernel Version

## yum list updates syntax

yum list updates {package-name}

yum list updates 'php\*'

# yum list updates 'kernel\*'

You can list **currently installed package version** by typing any one of the following command:  
# rpm -q kernel  
OR  
# yum list installed 'kernel\*'  
Or you can list multiple packages on separated by whitespace:  
# yum list installed 'kernel\*' 'php\*'

# yum update kernel.i686

Remove Old kernels

First, make sure yum-utils is installed:

yum install yum-utils

Next, check the kernels that you have installed:

rpm -q kernel

kernel-2.6.32-279.el6.x86\_64

kernel-2.6.32-279.2.1.el6.x86\_64

kernel-2.6.32-279.5.2.el6.x86\_64

kernel-2.6.32-279.9.1.el6.x86\_64

And finally to keep only two of the latest kernels, run:

package-cleanup --oldkernels --count=2

To make the change permanent when installing / updating kernels via yum, edit the **/etc/yum.conf** file and make the following change:

installonly\_limit=2

Yum Command Check and Apply Only Security Updates

## Install yum-plugin-security

# yum -y install yum-plugin-security

# yum -y install yum-plugin-security

# yum --security check-update

# yum updateinfo list bugzillas

1. **Another related setting in the virtual memory subsystem is the ratio at which dirty pages created by application disk writes will be flushed out to disk.**

The default value 10 means that data will be written into system memory until the file system cache has a size of 10% of the server’s RAM.

The ratio at which dirty pages are written to disk can be altered as follows to a setting of 20% of the system memory

**# sysctl -w vm.dirty\_ratio=20**

1.Booting procedure of linux server  
2.user administration  
3.LVM -FS creation/ extend / FS issues & recover  
4.Software install / remove / update - RPM or YUM   
5.Basic services config /troubleshoot - E.G : NFS ,FTP , SAMBA , DNS , DHCP   
6.Utilization issues - CPU , MEMORY , DISK   
7.Hardware errors and troubleshoot  
8.PAtch installation / roll back